INFORME DE TESTING Y PRUEBAS DE CÓDIGO

Las pruebas de software o testing requieren un personal con altos conocimientos y estudios sobre el tema para poder profesionalizarse.

El testing o pruebas de software es un paso muy relevante en cuanto al ciclo de desarrollo de un software. Ayudan a encontrar lo que son errores durante la etapa de desarrollo para que cuando dicho software entre en el mercado, el usuario/cliente tenga la mayor satisfacción posible en cuanto al uso de él. Ayuda a determinar su calidad antes de que entre en el mercado, así habilitando la entrada de clientes antes de tiempo e inversores.

En cualquier producto existe la posibilidad de que algo malo llegué a ocurrir, pero realizando dichas pruebas de software reducen estas posibilidades además de que, si ocurriesen, estaríamos más preparados para afrontar esa situación.

Pasar por este proceso no descarta al 100% que el software no tenga ningún tipo de error, pero como he mencionado anteriormente, si que ayuda posteriormente para la confrontación de problemas que puedan suceder en un futuro.

El testing que nosotros conocemos actualmente no es el que ha habido desde un principio, sino que ha habido una serie de avances tecnológicas permitiéndonos estar al nivel al que estamos.

El testing y las pruebas de software tienen diferencias, pero no son extremadamente notables.

La definición de testing es buscar, evaluar y encontrar errores en todo el proyecto. Utiliza técnicas diferentes que explicaré después cuando comente los diferentes tipos de pruebas y las diferentes técnicas de testing.

En cuanto a las pruebas de software realizan en principio lo mismo que el testing, pero en vez de buscar en general, buscan en lugares más específicos además de verificar la corrección que se ha realizado en el código fuente.

El testing tiene el objetivo de asegurar el uso cómodo del software por parte del usuario además de que cumpla sus expectativas, además los testing normalmente se pueden hacer en cualquier momento del ciclo de desarrollo.

En cuanto a las pruebas de software, su objetivo es conseguir la mejor calidad de software en cuanto a mantenimiento, visibilidad de él, y el cumplimiento de los estándares fijados.

Hay diferentes tipos de pruebas de software como:

- Pruebas funcionales, las cuales se centran en verificar que el software cumple requisitos especificados anteriormente. Dentro de las pruebas funcionales hay una variedad de pruebas: unitarias, de integración, de sistema y de aceptación entre otras.
- Pruebas no funcionales, verifican aspectos los cuales no son directamente vistos por el usuario, por ejemplo, el rendimiento, la seguridad…. Dentro de estas pruebas existen: las pruebas de rendimiento, de seguridad, de usabilidad, accesibilidad y compatibilidad entre otras.

Las técnicas más conocidas al realizar testing son TDD (Test Driven Development), BDD (Behaviour Driven Development).

El TDD es una forma de afrontar el código ya que se crean y se escriben los tests antes que el código. Esta técnica ofrece varias ventajas:

- Se asegura que todo el código nuevo que sea escrito reciba una prueba.
- Permite que el código este bien ordenado y documentado ya que cada test clasifica el propósito del código que ha testeado.
- Esta técnica permite una mayor productividad, dicho por diversos developers.
- Los softwares que pasan por esta técnica son mayoritariamente más estables.

Los contras de esta técnica son los siguientes:

- Algunos tests pueden no reconocer algunos fallos ya que el código que estaría testeando sería muy grande.
- Hay que escribir y mantener los tests.
- Se necesita tiempo utilizando esta técnica para volverte muy efectivo.
- A veces, la continua creación de tests te distrae de otras partes del software.

Hubo un estudio en 2005 el cual descubrió que al usar TDD es necesario estar escribiendo continuamente tests, pero estos programadores que escribían más tests llegaron a ser más productivos. La hipótesis sobre la calidad del código fue poco concluyente.

Ha llegado un punto en nuestra época donde todas estas pruebas que hemos estado nombrando han pasado a ser automatizadas. En cambio, no todas las pruebas pueden ser automatizadas, algunas tienen que ser realizadas por un humano.

Los proyectos que necesitan una entrega continua necesitan estas automatizaciones ya que realizan su trabajo mucho más rápido que un humano. Las pruebas que deben y son automatizadas son, por ejemplo:

- Las pruebas de extremo a extremo las cuales simulan una experiencia de un usuario en un software.
- Las pruebas unitarias abarcan unidades individuales de código.

En cambio, como he dicho antes, no todas las pruebas pueden ser automatizadas, aquí podemos observar un par de pruebas que solo pueden ser realizadas manualmente:

- Las pruebas exploratorias no pueden ser automatizadas ya que son muy aleatorias y prueban secuencias sin ningún tipo de script para encontrar fallos inesperados.
- Las pruebas de regresión visual se ejecutan cuando se introduce un defecto de diseño visual que puede ser vista por el usuario. Estas pruebas pueden ser automatizadas, pero llegan a ser tan costosas y tan poco efectivas que es mejor que las realice un humano.

En conclusión, los testings y las pruebas de software son vitales en cuanto al desarrollo de un software ya que sin ellos habría muchísimos problemas tanto para las empresas que realizan ese software que para los usuarios que llegarían a utilizarlo. La corrección de errores es fundamental para un buen funcionamiento y un agradable uso.

Referencias:

Consulting Group: Las consecuencias de no incluir pruebas de calidad en un proyecto de software [en línea], (sin fecha). Consulting Group. [Consultado el 22 de enero de 2024]. Disponible en: https://www.cgclatam.com/Blog/Las-consecuencias-de-no-incluir-pruebas-de-calidad-en-un-proyecto-de-software#:~:text=Entre%20los%20impactos%20que%20podrían,Pérdida%20de%20dinero.

Pruebas de Software: Historia y Evolución [en línea], (sin fecha). FYC Group: Servicio de Desarrollo y Pruebas de Software. [Consultado el 22 de enero de 2024]. Disponible en: https://www.fyccorp.com/articulo-pruebas-de-software:-historia-y-evolucion#:~:text=Según%20ISTQB,%20el%20testing%20de,producto%20y%20su%20correcto%20funcionamiento.

Testing software: Qué son las pruebas de software [en línea], (sin fecha). Epitech Spain. [Consultado el 22 de enero de 2024]. Disponible en: https://www.epitech-it.es/testing-software-pruebas-software/#:~:text=El%20testing%20de%20software%20es,de%20rendimiento,%20seguridad%20y%20funcionalidad.

García, G., (2021). Videotutorial Testing y prueba de código- Fundamentos esenciales de la programación | LinkedIn Learning, antes Lynda.com [en línea]. LinkedIn. [Consultado el 22 de enero de 2024]. Disponible en: https://es.linkedin.com/learning/fundamentos-esenciales-de-la-programacion-2/testing-y-prueba-de-codigo#:~:text=El%20testing%20o%20pruebas%20de,errores%20y%20mejorar%20su%20calidad.

Contributors to Wikimedia projects, (2003). Test-driven development- Wikipedia [en línea]. Wikipedia, the free encyclopedia. [Consultado el 22 de enero de 2024]. Disponible en: https://en.wikipedia.org/wiki/Test-driven_development#:~:text=Test%20Driven%20Development%20(TDD)%20is,leading%20to%20more%20robust%20software.

Pruebas de software automatizadas para la entrega continua [en línea], (sin fecha). Atlassian. [Consultado el 22 de enero de 2024]. Disponible en: https://www.atlassian.com/es/continuous-delivery/software-testing/automated-testing